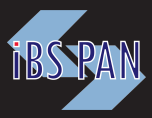

INSTYTUT BADAŃ SYSTEMOWYCH
POLSKIEJ AKADEMII NAUK

TECHNIKI INFORMACYJNE TEORIA I ZASTOSOWANIA

Wybrane problemy
Tom 4 (16)

poprzednio

ANALIZA SYSTEMOWA W FINANSACH
I ZARZĄDZANIU

Pod redakcją
Andrzeja MYŚLIŃSKIEGO

Warszawa 2014

INSTYTUT BADAŃ SYSTEMOWYCH
POLSKIEJ AKADEMII NAUK

TECHNIKI INFORMACYJNE TEORIA I ZASTOSOWANIA

Wybrane problemy
Tom 4 (16)

poprzednio

ANALIZA SYSTEMOWA W FINANSACH I ZARZĄDZANIU

Pod redakcją
Andrzeja Myślińskiego

Warszawa 2014

Wykaz opiniodawców artykułów zamieszczonych
w niniejszym tomie:

Prof. Bernard De BAETS

Dr hab. Ewa BEDNARCZUK, prof. PAN

Dr hab. inż. Wiesław KRAJEWSKI, prof. PAN

Dr hab. inż. Andrzej MYŚLIŃSKI, prof. PAN

Dr inż. Jan W. OWSIŃSKI

Dr hab. Dominik ŚLĘZAK, prof. UW

Prof. dr hab. inż. Andrzej STRASZAK

Prof. dr hab. inż. Stanisław WALUKIEWICZ

ISBN 83-894-7555-3

ANALIZA WPŁYWU KOMPRESJI NA PRZYSPIESZENIE ŁADOWANIA STRON WWW

Tomasz Janczewski

Instytut Badań Systemowych Polskiej Akademii Nauk,
Studia Doktoranckie, Warszawa, Polska,
Tomasz.Janczewski@ibspan.waw.pl

Streszczenie. W pracy przeanalizowano zasadność użycia kompresji zasobów HTTP oraz jej wpływ na czas ładowania strony WWW. W literaturze można znaleźć wiele prac poświęconych tematyce szybkości ładowania stron WWW, jednak nie wszystkie wymienione działania faktycznie przyspieszają ładowanie stron. Typowym przykładem jest użycie kompresji gzip dla wszystkich zasobów przesyłanych w ramach witryny WWW. Jest to mało skuteczne rozwiązanie w przypadku plików gif lub jpeg.
Słowa kluczowe: WWW, kompresja, format plików, Internet, protokół TCP/IP

1 WSTĘP

Podstawą działania sieci WWW (World Wide Web) jest protokół HTTP (Hypertekst Transfer Protocol). Dział a on na zasadzie przesyłania żądania i odbierania odpowiedzi (ang. Request-Response). Protokół jest bezstanowy – oznacza to, że każde żądanie strony jest odrębną transakcją, realizowaną niezależnie od poprzednich żądań, nawet gdyby dotyczyły one stron wchodzących w skład tej samej aplikacji [6]. Konsekwencją takiego sposobu funkcjonowania HTTP jest konieczność implementowania stanowości w aplikacjach wykorzystujących ten protokół. Obecnie protokół HTTP stanowi podstawę komunikacji pomiędzy przeglądarką użytkownika, a serwerami dostarczającymi treści stron WWW.

Celem niniejszego artykułu jest rozważenie celowości używania kompresji gzip do każdego rodzaju zasobu przesyłanego użytkownikowi w ramach portalu internetowego.

Pliki stron WWW zwykle są opisywane przez język znaczników HTML (Hypertext Markup Language). Każdy transfer możemy opisać jako:

1. wywołanie żądania,
2. odnalezienie zasobu na serwerze,
3. przesłanie odpowiedzi.

Tworzenie zapytania i jego weryfikacja odbywa się po stronie przeglądarki internetowej, natomiast obsługa realizowana jest poprzez serwer WWW lub serwer CDN (ang. Content Delivery Network). Serwery CDN znajdują

się w różnych częściach internetu i są odpowiedzialne za dostarczanie ciężkich zasobów, takich jak na przykład video dla użytkowników znajdujących się w zasięgu danej sieci.

W dalszej części pracy wykorzystywane będą dwa pojęcia: efektywność przesyłania informacji oraz czas odpowiedzi. Efektywność jest mierzona liczbą jednostek danych, wyrażoną w bajtach, możliwą do przesłania w danej jednostce czasu wyrażonej w sekundach. Celem przyjętym w tej pracy jest maksymalizacja efektywności przesyłania zasobów portalu internetowego. Czas odpowiedzi (ang. response time) to łączny czas, jaki upływa od momentu utworzenia żądania do chwili pełnego załadowania strony WWW przez przeglądarkę internetową. Efektywność jest odwrotnie proporcjonalna do czasu odpowiedzi.

2 PROTOKÓŁ HTTP

Protokół HTTP umożliwia oglądanie stron, przesyłanie informacji za pomocą formularzy, wyświetlanie filmów itd. Istnieją dwie powszechnie wykorzystywane wersje protokołu HTTP: HTTP/1.0 oraz HTTP/1.1.

Działa on w oparciu o zasadę żądanie–odpowiedź, przy czym protokół HTTP jako taki nie przechowuje informacji o poprzednim wywołaniu. Transakcje w HTTP są ograniczone do pary żądanie-odpowiedź. Każde wywołanie oraz odpowiedź wykorzystuje protokół TCP.

Protokół HTTP/1.1 w przeciwieństwie do poprzednich wersji utrzymuje jedno stałe połączenie pomiędzy maszyną użytkownika a serwerem WWW [3]. Ma to wpływ na efektywność wykorzystania zasobów serwera oraz komputera użytkownika poprzez eliminację czynności związanych z nawiązaniem połączenia, takich jak na przykład wywołanie DNS. Zmiana ta pozwala zaoszczędzić czas potrzebny na zainicjowanie połączenia oraz na komunikację z serwerami DNS. Przeglądarki WWW różnie implementują sposób pobrania zasobu strony. Mogą działać według następujących schematów:

- wysyłać kolejne żądanie dopiero po otrzymaniu odpowiedzi na poprzednie żądanie,
- wysłać wszystkie żądania równolegle, np. w oddzielnych wątkach i czekać na odpowiedzi.

Wysyłanie równoległe żądań oraz równoległe przetwarzanie odpowiedzi skraca czas ładowania strony WWW. Fakt równoległego przetwarzania żądań nie poprawia efektywności, jednak znacząco zmniejsza czas obsługi żądania/odpowiedzi po stronie przeglądarki [1]. Wynika to z faktu braku

zmiany całkowitej liczby jednostek danych możliwych do przesłania w danej jednostce czasu.

Schematycznie transakcję HTTP rozumianą jako pojedynczą atomową jednostkę komunikacji można przedstawić następująco:

Rys. 1. Schemat ogólny komunikacji HTTP

W komunikacji udział biorą: przeglądarka internetowa, serwer WWW, serwer DNS, opcjonalnie serwery CDN. Wyszczególniamy następujące fazy dostarczenia zasobu WWW do użytkownika:

1. nawiązanie połączenia,
2. wysłanie żądania,
3. realizacja żądania,
4. wysłanie odpowiedzi,
5. renderowanie strony.

3 ANALIZA KOMUNIKACJI HTTP

W celu analizy komunikacji przy użyciu protokołu HTTP przeprowadzono badanie polegające na obserwacji żądań w trakcie przeglądania zasobów WWW portalu Wirtualna Polska (www.wp.pl) przez określony z góry czas. Do obserwacji żądań HTTP użyto programu Whireshark [2], który posiada filtry pozwalające dokonywać analizy zarówno żądań, jak i odpowiedzi.

Odpowiedzi zwracane przez serwer WWW miały postać plików, które w kolejnym kroku eksperymentu zostały poddane porównaniu względem wielkości jako wersje spakowane (gzip) oraz niespakowane. Wyniki prezentowane są w Tabeli 3. Obserwacji dokonano przy następujących parametrach łącza komputera, na którym uruchomiona była przeglądarka:

– ruch przychodzący (download) : 20Mbps,
– ruch wychodzący (upload) : 2Mbps.

3.1 Zbadana próba

Obserwacji poddano 2129 wywołań żądań dostarczenia zasobów. Średnia wielkość żądania oraz odpowiedzi wyniosła 720 kilobajtów, jednak należy brać pod uwagę, że w badanej grupie były zarówno małe pliki tekstowe zawierające kilka kilobajtów, jak również duże pliki graficzne zawierające kilkanaście megabajtów.

3.2 Zasoby

Jako zasób rozumiemy część strony WWW przesyłaną w ramach oddzielnej transakcji.

Tabela 1 przedstawia liczbę żądań dostarczenia zasobów dla różnych wykorzystywanych metod oraz liczbę odpowiedzi o danym statusie.

Tabela 1. Liczba wywołań poszczególnych metod protokołu HTTP

Metoda HTTP	Liczba wywołań
GET	2044
POST	85
Odpowiedzi serwera zgodnie z protokołem HTTP	
2xx: Success	1932
3xx: Redirection	193
4xx: Client Error	4

Z obserwacji wynika, że większość żądań jest realizowana za pomocą metody GET, co z kolei wskazuje na stosowanie technologii asynchronicznej java script i xml. AJAX znacząco skraca czas odpowiedzi poprzez przesyłanie jedynie fragmentów strony WWW w celu renderowania przez przeglądarkę, zamiast całości dokumentu. Przedstawione w Tabeli 1 dane pokazują również, że prawie wszystkie żądania zakończyły się sukcesem. Do ich niezrealizowania doszło jedynie w czterech spośród 2129 przykładów.

Serwery Apache nie logują pełnych wywołań POST, w przeciwieństwie do wywołań GET [4], które są logowane do plików access.log w całości. W przypadku pojedynczych żądań czasy zapisu nie mają znaczenia, jednak w przypadku serwerów z dużą liczbą użytkowników, jednocześnie odpytujących serwer, lepiej jest używać wywołań metody POST, właśnie w celu ograniczenia nadmiarowej pracy dysku.

Tabela 2 pokazuje podział odpowiedzi ze względu na typ oraz średnie wielkości przesyłanego zasobu danego typu wyrażone w bajtach.

Tabela 2. Podział par żądanie–odpowiedź ze względu na typy przesyłanych danych

Typ zasobu	średnia wielkość przesyłanego zasobu [B]
text/html	16874
text/css	64575
application/javascript	28367
application/x-shockwave-flash	30438
application/x-font-woff	19640
application/xml	360
audio/mpeg	861203
image/jpeg	25628
image/png	6485
image/x-icon	10782
image/gif	4589

Analizując wyniki z Tabeli 2 można zobaczyć, iż największe średnie wielkości zasobów wiążą się z przesyłaniem filmów: audio/mpeg - 861203 byte; pliki css - 64575 byte oraz pliki flash - 30438 byte. Natomiast średnia wielkość przesyłanego zasobu graficznego to 11871 byte. Jako zasób graficzny traktujemy: image/gif, image/x-icon,image/png oraz image/jpeg.

W przypadku poprawy czasu odpowiedzi strony należy skupić się na optymalizacji wielkości przesyłanych obiektów związanych z mediami (jak filmy czy flash) oraz plikach tekstowych CSS, poprzez użycie kompresji danych. Dodatkowym czynnikiem może tu być użycie serwerów CDN dla zasobów graficznych, co wpłynie na zmniejszenie obciążenia serwerów dostarczających dane (mniej żądań do obsłużenia przez pierwotny serwer) oraz skrócenia drogi, jaką muszą przebywać pakiety względem pierwotnego serwera WWW.

3.3 Nagłówki

Każde z żądań posiada sekcję nagłówków oraz sekcję danych. Przykładowe wywołanie metody protokołu HTTP wygląda następująco:

GET http://www.wp.pl HTTP/1.1\r\n
Host: www.wp.pl\r\n
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:27.0)
Gecko/20100101
Firefox/27.0\r\n
Accept: text/html, application/xhtml+xml, application/xml; q=0.9,*/*;
q=0.8\r\n
Accept-Language: pl, en-us; q=0.7, en; q=0.3\r\n

Accept-Encoding: gzip, deflate\r\n
Connection: keep-alive\r\n

Przykładowa odpowiedź z serwera na zadane żądanie wygląda następująco:

HTTP/1.1 200 OK\r\n
Date: Mon, 10 Mar 2014 15:03:59 GMT\r\n
Server: Apache/2.2.0 (Fedora)\r\n
Expires: Thu, 19 Nov 1981 08:52:00 GMT\r\n
Content-Encoding: gzip\r\n
Content-Length: 4985\r\n
Content-Type: text/html; charset=UTF-8\r\n
\r\n

[... C O N T E N T ...]

Nagłówki są częścią dodatkowego nakładu danych na każde z żądań, co zmniejsza efektywność przesyłania danych zasobu. Należy zwrócić uwagę na stałość większości nagłówków w ramach jednej sesji. Oczywiście zmienia się np. pierwsza linia w nagłówkach żądania GET, czy też nagłówek Content-Length w odpowiedzi, jednak znacząca większość pozostaje niezmienna.

Należałoby zastanowić się nad sensem przesyłania wszystkich nagłówków przy każdym żądaniu w ramach jednej sesji. Przeszkodą jednak jest tu brak stanowości protokołu HTTP oraz różne podejście do renderowania stosowane przez przeglądarki.

3.4 Kompresja

Wszystkie obserwowane odpowiedzi były automatycznie poddawane kompresji gzip. Zapewnia ona znakomite wyniki w przypadku zasobów tekstowych lub plików ICO, jednak stosowanie owej kompresji do każdego rodzaju zasobu może powodować straty efektywności.

Tabela 3 przedstawia analizę wybranych z badanej próby przykładowych zasobów. W badanej próbie prawie połowa wywołań (984), to wywołania plików graficznych, które zostały poddane kompresji gzip, tak jak wszystkie inne zasoby. Tabela 3 wskazuje, że niektóre zasoby są większe po kompresji niż przed nią, gdy procent kompresji był równy 0. Przykładem mogą być pliki mso, gif, png (procent kompresji równy 1).

Sama operacja kompresji po stronie serwera i dekompresji po stronie przeglądarki powoduje większe zużycie zasobów RAM, czasu pracy procesora (kolumna numer 3 w Tabeli 3) oraz dysku twardego. Czas zużycia

Tabela 3. Analiza przesłanych zasobów

Typ zasobu	Oryginalny rozmiar [B]	Rozmiar po kompresji GZIP [B]	Czas pracy procesora podczas kompresji /dekompresji [ms]	Ilość skompresowanych danych, wynik kompresji [B]	Procent kompresji [%]
Plik mht	100762	100762	7	79571	-79
Plik bmp	587934	475784	75	112150	-20
Plik docx	15713	13157	3	2556	-17
Plik pdf	225663	222149	23	3514	-2
Plik exe	844267	816975	84	27292	-4
Plik xml	314	212	2	102	-33
Plik mov	58213474	54727381	5944	3486093	-6
Plik jpg	827217	815648	60	11569	-2
Plik mso	2749	2772	2	-23	0
Plik gif	1231	1254	2	-23	0
Plik jar	142714	132425	13	10289	-8
Plik ppt	6488576	6268141	657	220435	-4
Plik xsl	152064	42685	19	109379	-72
Plik odp	8539899	8406542	937	133357	-2
Plik tif	8055690	5318169	1158	2737521	-34
Plik png	1763408	1759575	172	3833	-1
Plik mpg	179988484	159232161	20976	20756323	-12
Plik doc	99328	80544	13	18784	-19
Plik thmx	3092	2744	2	348	-12

procesora potrzebny na kompresję /dekompresję może być nieadekwatnie długi w stosunku do wyniku kompresji. Widzimy taki przypadek podczas kompresowania plików mov, mpg lub tif (Tabela 3). W skrajnym przypadku strony WWW stworzonej z samych plików gif oraz mpg możemy uzyskać wyłącznie straty efektywności przesłania całości zasobów, spowodowane przez kompresję GZIP. Idealnym rozwiązaniem byłoby tu użycie dynamicznego wyłączania kompresji dla zasobów graficznych, np. przy pomocy algorytmów uczących się.

4 WNIOSKI

Nawet krótkie odwiedziny portalu internetowego generują dość spory ruch HTTP, który należy optymalizować. Pojedyncza sesja nie obciąża mocno zasobów serwera, jednak zwielokrotnienie liczby użytkowników, przy niewłaściwej optymalizacji portali lub jej braku, może doprowadzić do zablokowania ruchu.

Podczas optymalizacji należy skupić się na jej sensowności. Kompresowanie zawartości strony składającej się tylko z plików graficznych gif wydaje się nieuzasadnione, gdyż może doprowadzić w skrajnym przypadku nawet do 30-procentowego spadku efektywności przesyłania strony WWW. Dobrym rozwiązaniem, w przypadku stron o bogatej treści medialnej, jest stosowanie serwerów CDN zlokalizowanych bliżej użytkownika końcowego, które są w stanie przejąć na siebie część ruchu związanego z mediami.

Projektując witrynę WWW, należy zwrócić uwagę na sposób wywoływania żądań, gdyż żądania GET logowane są po stronie serwera w całości, w przeciwieństwie do żądań typu POST. W przypadku dużej liczby użytkowników oraz długiego czasu przechowywania plików log, wiąże się to z kosztami przestrzeni dyskowej oraz stratami efektywności związanymi z zapisem na dysku twardym serwera. Same przeglądarki również można poddać optymalizacji, by nie przesyłać w każdym żądaniu niezmienionych nagłówków.

LITERATURA

1. Sathya Narayanan Nagarajan, Srijith Ravikumar (2012) *Model for Predicting End User Web Page Response Time*. http://arxiv.org
2. *Wireshark network protocol analyzer* (2014) http://www.wireshark.org/
3. *W3C. Hypertext Transfer Protocol – HTTP/1.1 Spec.* (1999). The Internet Society.
4. *Apache HTTP Server Version 2.4 Documentation*(2014). http://httpd.apache.org/docs/2.4/
5. Billy Hoffman (2013) *Lose the Wait: HTTP Compression 2012*. http://zoompf.com/blog/ 2012/02/lose-the-wait-http-compression
6. *Bezpieczeństwo serwisów WWW* (2014) http://webmaster.helion.pl /index.php/php

COMPRESSION IMPACT ANALYSIS ON THE ACCELERATION OF WWW PAGES LOADING

Abstract. The paper deals with the analysis of the rational use of HTTP resources compression and its impact on the WWW pages loading time. In literature, there are many papers dealing with the speed of the WWW pages loading. However, many of the proposed methods, when examined,

do not really speed up WWW pages loading. The known example of the low efficiency of WWW pages loading is application of gzip compression to all the resources sent within one WWW page. Such an approach is also not effective in the case of gif or jpeg files.
Keywords: WWW, compression, files format, Internet, TCP/IP protocol

ISBN 83-894-7555-3